Defence Research and Development Canada
Recherche et développement pour la défense Canada

DEFENCE R&D DÉFENSE

Field Trial data Analysis and Testing (FiTAT) tool

Mathieu Caillet

Defence R&D Canada – Ottawa

Contract Report
DRDC Ottawa CR 2011-143
October 2011

Canada

Field Trial data Analysis and Testing (FiTAT) tool

Mathieu Caillet

Prepared by:

Royal Military College of Canada
PO Box 17000, Station Forces
Kingston, Ontario, Canada K7K 7B4

Project Manager: Mike Vinnins
Contract Number: A1410FE255
Contract Scientific Authority: Michel Clénet

Defence R&D Canada - Ottawa
Contract Report
DRDC Ottawa CR 2011-143
October 2011

Scientific authority

Original signed by Michel Clénet

Michel Clénet

Approved by

Original signed by Bill Katsube

Bill Katsube
SH CNEW

Approved for release by

Original signed by Chris McMillan

Chris McMillan
Chairman DRP

Abstract

This report presents the Field Trial data Analysis and Testing (FiTAT) tool developed at DRDC Ottawa to analyze and process acquired data files recorded during field trials. Using FiTAT, studies can be conducted to select the appropriate algorithms and hardware meeting cost and performance. The implementation of the FiTAT tool and a user guide are described here to provide useful information to software developers as well as users. In addition, parallel computing is available to reduce the processing time of very large data files.

Résumé

Le présent document présente l'outil Field Trial data Analysis and Testing (FiTAT) développé à DRDC Ottawa pour l'analyse et le traitement de fichiers de données acquises au cours de campagne de mesures. À l'aide de FiTAT, des études peuvent être menées afin de sélectionner les algorithmes appropriés et les composants du système satisfaisants aux coût et performances. L'implémentation de l'outil FiTAT ainsi qu'un guide d'utilisation sont décrits ici pour fournir les informations utiles aux développeurs et aux utilisateurs. De plus, le calcul en parallèle a été considéré afin de réduire le temps de traitement des fichiers de données volumineux.

This page intentionally left blank.

Executive summary

Field Trial data Analysis and Testing (FiTAT) tool

Mathieu Caillet; DRDC Ottawa CR 2011-143; Defence R&D Canada - Ottawa; October 2011.

Background: Two data acquisition systems are currently available in the navigation group in the Communication and Navigation Electronic Warfare (CNEW) section at DRDC Ottawa. Data can be recorded during field trials to test and evaluate components of GPS/GNSS systems (specifically anti-jam systems), or analyze frequency spectrum through post-processing. In order to perform the evaluation of the performance, it was required to implement a program tool which is able to read the acquired data files and process the data.

Principal results: The implementation of a graphical user interface (GUI) has been achieved to facilitate the processing of the acquired data files. It allows to plot the acquired or processed data, and compute data file processing using various methods. Parallel computing and batch run are available to reduce the processing time and to increase the efficiency.

Significance of results: Several nulling methods have been implemented to process the acquired data. The FiTAT tool helps along in the evaluation of the performance for each of the configurations tested during the field trials. After a method has been identified to meet the targeted performance, the required hardware would be chosen.

Future work: The next step would consist in making possible the inter-operability of FiTAT and another tool called PAASoM that was developed at DRDC Ottawa. In particular, the extraction of the steering vector contained in the acquired data could be carried out in Fitat and used in PAASoM to determine the achievable radiation characteristics. Also, the implementation of algorithms for direction finding (with or without calibration) is considered.

Sommaire

Field Trial data Analysis and Testing (FiTAT) tool

Mathieu Caillet ; DRDC Ottawa CR 2011-143 ; R & D pour la défense Canada - Ottawa ; octobre 2011.

Contexte : Deux systèmes d'acquisition de données sont présentement disponibles au sein du Groupe navigation dans la section Guerre électronique (Communication et navigation) à RDDC Ottawa. Des données peuvent être acquises pendant les campagnes de mesures pour tester et évaluer les différents composants des systèmes GPS/GNSS, ou analyser les spectres fréquenciels par post traitement. Dans le but d'évaluer et d'analyser les performances, il était nécessaire d'implémenter un outil informatique capable de lire les fichiers de données acquises et de traiter ces données.

Résultats : L'implémentation d'une interface graphique a été réalisée pour faciliter le traitement des fichers de données acquises. Cette interface permet d'afficher les données acquises ou traitées, et de procéder au traitement des fichiers de données en utilisant diverses méthodes. Le calcul parallèle et séquentiel sont disponibles pour réduire le temps de calcul et être plus efficace.

Portée des résultats : Plusieurs méthodes ont été implémentées pour traiter les données acquises. L'outil FiTAT aide dans l'évaluation des performances de chacune des mesures réalisées sur le terrain. En particulier, l'identification d'une méthode satisfisant les performances d'un système permet de sélectionné les composants du système à concevoir.

Recherches futures : La prochaine étape consiste à rendre possible l'inter-opérabilité de FiTAT avec un autre outil informatique nommé PAASoM qui a été développé à RDDC Ottawa. En particulier, l'extraction du vecteur source (amplitude/phase pour chaque antenne du réseau) contenu dans les données acquises pourrait être faite par FiTAT et utilisé dans PAASoM pour déterminer les caractéristiques de rayonnement.

Table of contents

List of figures

Acknowledgements

The author would like to thank:
– Ambighairajah Yasotharan for contributing to the implementation of some algorithms;
– Collin Wilson for providing sample data acquired using the Gage card;
– Scott McLelland for providing sample data recorded with the DTA system along with the Matlab code allowing to retrieve the data of each channel;
– Michel Clénet for sharing ideas and initial Matlab code to process the recorded data.

1 Introduction

Data acquisition systems (DAS) are very useful to test different signal processing algorithms using collected data before settling down on a hardware architecture. Using DAS allows one to have access to various parameters and identify the critical ones for the targeted application. Studies can then be conducted to select the appropriate hardware meeting cost and performance. In order to process the acquired data, it is first required to read the data and then to arrange it in a format suitable for digital computation. If multiple configurations and algorithms are tested, several routines will most likely be used to process the data. In this case, a user friendly graphical interface is convenient to integrate those routines in the same environment.

Two DAS are currently available in the Navigation group in the Communication and Navigation Electronic Warfare (CNEW) section at DRDC Ottawa. Those devices are a Gage Octopus 8-channel 14-bit card [1], and a D-TA 8-channel 16-bit system [2]. Using both devices, a large volume of data have been acquired over various field trials [3, 4, 5]. A software tool, called Field Trial data Analysis and Testing (FiTAT), has been created to analyze the data of both DAS and process the data using nulling algorithms. The objective of this report is to present the design and implementation of the Fitat tool.

The different goals to be achieved in FiTAT are:
- Read data from:
  - Gage system;
  - DTA system;
  - raw data files.
- Save input data for use with the navigation software (GSNRx) developed by the PLAN group at the University of Calgary [6]. Input data files can have header or not;
- Plot input data versus frequency and/or time;
- Process input data using one of the available nulling algorithms;
- Plot output data versus Frequency and versus Time;
- Save output data:
  - In a format suitable for use with GSNRx;
  - In DTA format to allow playback;

This report is intended to provide details to the software developers of FiTAT as well as to the users. The report starts with the description of the Graphical User Interface (GUI) and the implementation of the code. This information is useful for software developers who may want to improve FiTAT, or add new features. A user guide is proposed at the end to assist users with the practical procedures of FiTAT.

Section 2 focuses on the implementation of the GUI, with the associated actions. The data handling is described in Section 3. Details regarding how data are converted to matrices are provided. Moreover, the structuring of the code and the implementation of the data pro-

cessing routines is presented in Section 4. In addition, a user guide is provided in Section 5. Finally, this reports gives some conclusions and perspectives.

2 Graphical User Interface (GUI)

This section describes the implementation of the GUI developped in Matlab [7]. The layout of the main window is described first, and then the different variables of the GUI are detailed. Finally, the main actions in the GUI are explained.

2.1 Description of the main window layout

The GUIDE (GUI Design Environment) module of Matlab has been used to accomplish the FiTAT GUI. In the Matlab environment, two files are required to implemented a GUI: a Figure file ('.fig'), and a Matlab file ('.m'). The Figure file contains the information regarding the different graphical components, and the Matlab file contains the code to be executed depending on the action taken in the GUI. Using GUIDE, the components can be arranged and saved in a '.fig' file. Function headers corresponding to the created GUI components are automatically generated and saved in the '.m' file.

The layout of the main window is shown in Fig. 1. On the top, the data type, the generic name of the data files, the channel names and the reference channel can be chosen. The available data types are 'Gage','DTA','Real raw data' and 'IQ raw data'. The data file generic name is defined as the common string of the data filenames under consideration. The varying string of the data filenames is replaced by 'XX'. The varying strings are entered one after the other separated by a comma in the channel name field. For the reference channel, the possible choices reflect the entered values in the 'Specify channel name(s)' field. In the middle area, the desired testing or processing are accessible with the corresponding parameters. The plotting options and printing are located at the low level. At the very bottom, three buttons can be pushed to either close or clear the GUI, and execute the current project.

A command menu is also available to open and save project files, and access the help. The menu has been created in GUIDE. Fig. 2 shows the command menu.

2.2 Variables of the GUI

The names of the components of the GUI are shown in Fig. 3. The names of the corresponding functions in the '.m' file are based on the names of the components. For instance, the code that achieves the actions related to the 'multiProc' checkbox will be implemented in the 'multiProc_Callback' function.

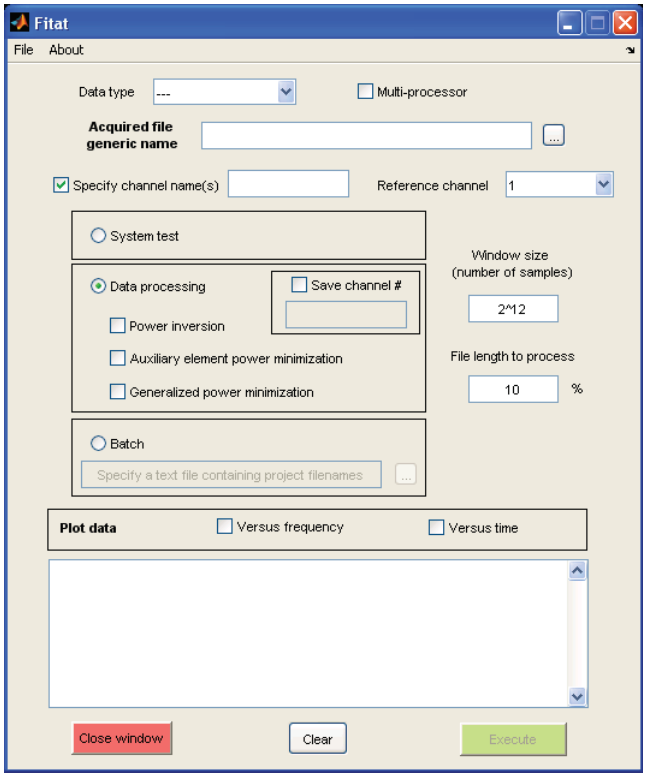

Figure 1: *FiTAT main window.*

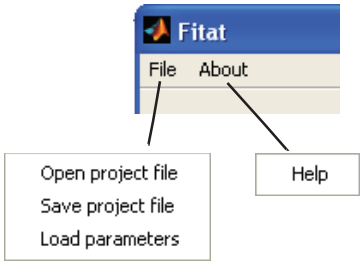

Figure 2: *FiTAT command menu.*

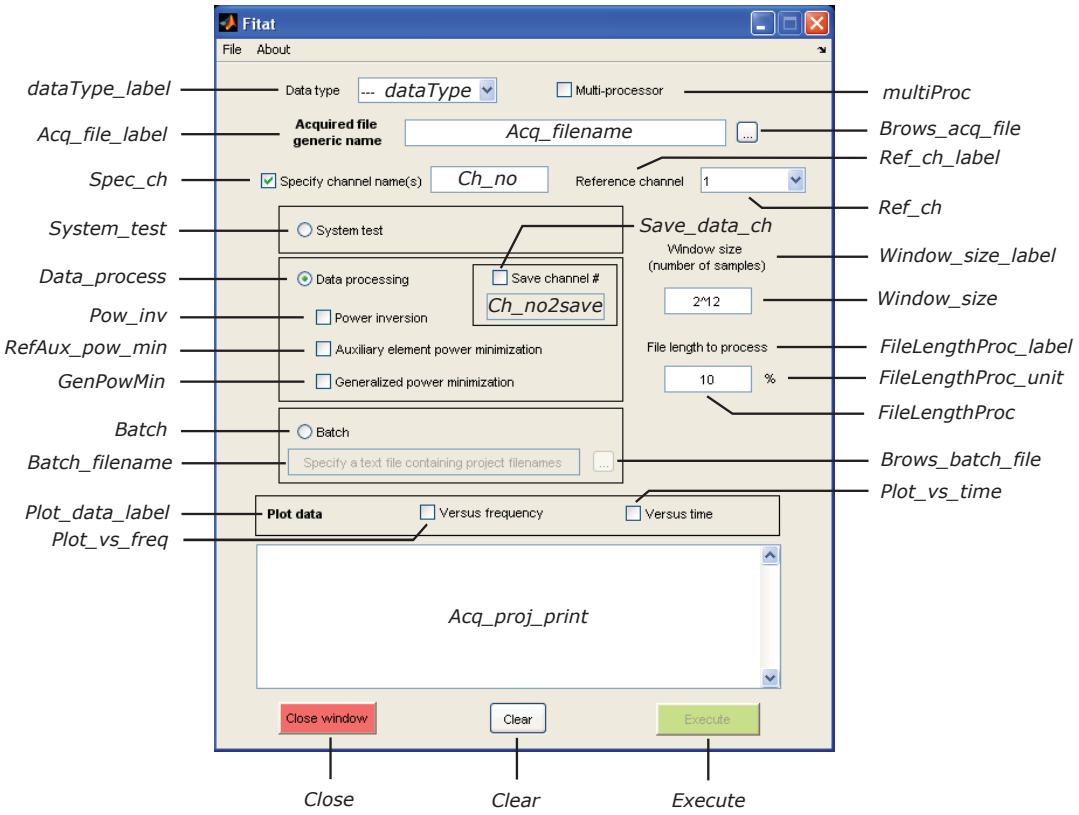

Figure 3: *FiTAT GUI variables.*

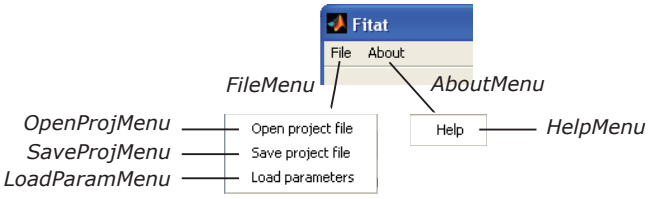

Figure 4: *FiTAT command menu variables.*

2.3 GUI actions

The actions carried out by each component of the GUI are described in this section. The list of the different graphical components is given below (from top to bottom of the FiTAT window) with a description of the actions and the interactions with other graphical components. Actions are achieved by adding the proper code in the GUI '.m' file. A Matlab GUI file is organised by functions corresponding to the graphical components. It is then easy to identify where the code should be added.

- 'dataType': no specific action carried out.

- 'multiProc': no specific action carried out.

- 'Acq_filename': Checks if the entered string corresponds to a valid filename. The field is cleared if the filename does not exist, and an error message is returned into the 'Acq_proj_print' field.

- 'Brows_acq_file': Opens a standard dialog box for retrieving a data file. The files are filtered according to the selected data type. If a previous file has been entered, the default directory is the corresponding filepath. Otherwise, the default directory is the FiTAT directory. A test is performed to make sure the retrieved file is valid. The field is cleared if the filename does not exist, and an error message is returned into the 'Acq_proj_print' field.

- 'Spec_ch': Enables 'Ch_no' if checked.

- 'Ch_no': Reflects the entered names into 'Ref_ch' if the string is not empty.

- 'Ref_ch': no specific action carried out.

- 'System_test': If selected, disables and unselect/uncheck 'Data_process', 'Pow_inv', 'RefAux_pow_min', 'GenPowMin', 'Save_data_ch' and 'Batch'. 'Ch_no2save' and 'Batch_filename' are cleared and 'Brows_batch_file' is disabled.

- 'Data_process': If selected, disables and unselect/uncheck 'System_test' and 'Batch', and enables 'Pow_inv', 'RefAux_pow_min', 'GenPowMin' and 'Save_data_ch'. 'Batch_filename' is cleared and 'Brows_batch_file' is disabled.

- 'Pow_inv': If checked, 'RefAux_pow_min' and 'GenPowMin' are unchecked.

- 'RefAux_pow_min': If checked, 'Pow_inv' and 'GenPowMin' are unchecked.

- 'GenPowMin': If checked, 'Pow_inv' and 'RefAux_pow_min' are unchecked.

- 'Save_data_ch': If selected, enables 'Ch_no2save'.

- 'Ch_no2save': no specific action carried out.

- ‘Batch’: If selected, disables and unselect/uncheck ‘System_test’, ‘Data_process’, ‘Pow_inv’, ‘RefAux_pow_min’, ‘GenPowMin’ and ‘Save_data_ch’, and enables ‘Batch_filename’ and ‘Brows_batch_file’. ‘Ch_no2save’ is cleared.

- ‘Batch_filename’: Checks if the entered string corresponds to a valid filename. The field is cleared if the filename does not exist, and an error message is returned into the ‘Acq_proj_print’ field.

- ‘Brows_batch_file’: Opens a standard dialog box for retrieving a batch file (‘.txt’). If a previous file has been entered, the default directory is the corresponding filepath. Otherwise, the default directory is the FiTAT directory. Check if the retrieved file is valid. The field is cleared if the filename does not exist, and an error message is returned into the ‘Acq_proj_print’ field.

- ‘Window_size’: Checks if the entered value is an integer, returns an error message into the ‘Acq_proj_print’ field if not.

- ‘FileLengthProc’: Checks if the entered value is an integer comprised between 0 and 100, returns an error message into the ‘Acq_proj_print’ field if not.

- ‘Plot_vs_freq’: no specific action carried out.

- ‘Plot_vs_time’: no specific action carried out.

- ‘Close’: Opens a confirmation dialog box to make sure the closing request is intended. If ‘Yes’ is pushed, a second dialog box pops up to allow for saving the current project. Else the confirmation dialog box is closed and it returns to the main window.

- ‘Clear’: Unselects/unchecks, disables and sets to the default values all the graphical components accordingly to create a new project. All the strings are erased from ‘Acq_proj_print’. If plots exist, they are closed.

- ‘Execute’: In the project mode, all the values of the project are read and saved in a ‘structure’ variable. The requested processing is then carried out. In the batch mode, the batch file is read and the requested projects are executed one after the other as in the project mode.

A function called ‘proj_req_data’ is used to verify that the data required for the project setup have been entered. This function is implemented into the GUI ‘.m’ file and it is used to enable or disable the ‘Execute’ button.

3 Program operation and data handling

This section presents how the FiTAT tool is implemented and how the data are managed. The management of the acquisition system data are described first, and then the different modes of operation of the program are detailed. Finally, the variables and functions used to save and retrieved the data from the GUI is reviewed.

3.1 Acquisition system data management

Data files recorded through DAS are generally very large. As a result, the processing has to be done sequentially using blocks of data. To speed up the computation, the processing can be carried out over several cores of the processor unit. To maintain the sequentiality, the data are distributed in such a way that two consecutive cores computes two consecutive blocks of data. In this fashion, the data can be retrieved in the proper order when the processing is done. The data splitting is shown in Fig. 5. A header of size *HS* and containing the acquisition parameters (for instance, sampling rate, number of samples, etc) is usually present at the beginning of the data file. The data ($file\_length - HS$) are located behind the header, and have to be split in block of data of length *Window_size* and possibly distributed across p cores of the processor unit. The required number of iteration to process the entire file is $m = (file\_length - HS)/(p * Window\_size)$, and it should be determined before starting the processing. Usually, the samples at the very end of the file cannot be processed, their number depends of the window size and the number of processors.

The data format of the Gage and DTA systems are different. Acquired data using the Gage card are stored in individual files, one file corresponding to one channel. In the case of the DTA system, the acquired data are stored on a Redundant Array of Independent Disks (RAID) array. There is no redundancy between the disks in the used setup. Because the data of a specific channel are stored across multiple disks and interleaved, it is necessary to re-construct the data for each channel before carrying out the processing. A routine has been implemented in Matlab to read the read the different parts of one channel and save it in one file [8]. This operation is time-consuming for long acquisitions.

3.2 Program operation

FiTAT can operate according two different modes of operation, plus the batch run:
- **System test** is used to either strip acquired data files (i.e. remove header) or plot data (either acquired or computed data);
- **Data processing** computes the output data using the selected method, optionally strip the acquired data. In this mode of operation, data cannot be plotted because it is potentially distributed among multiple processors;

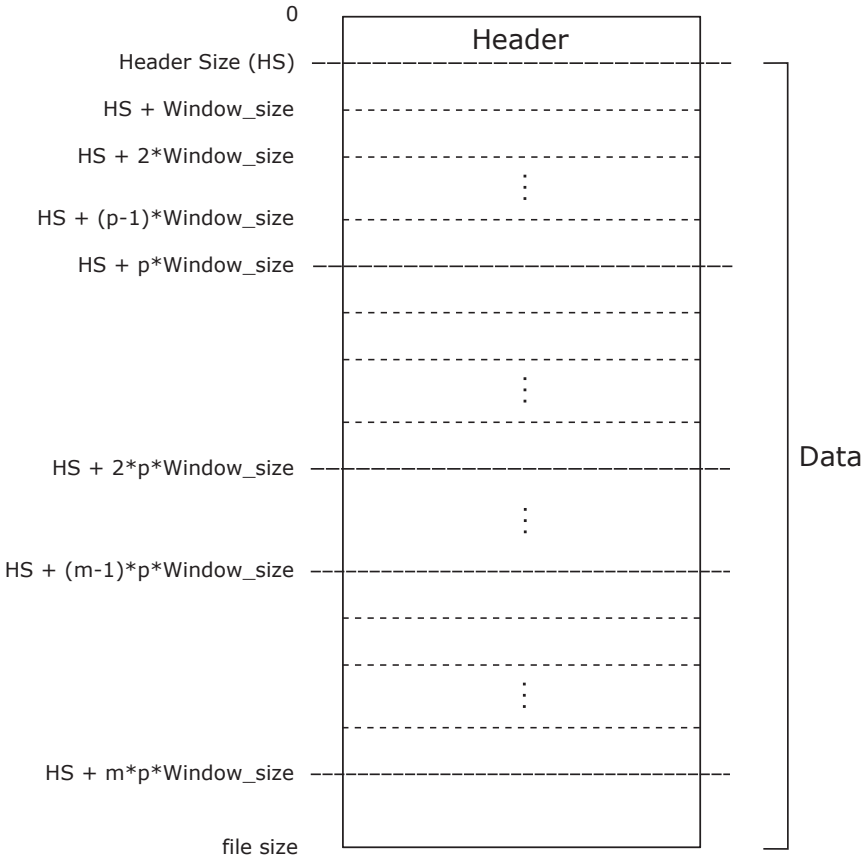

Figure 5: *Acquisition data splitting. Window_size is the length of the block of data being processed at each iteration, and p is the number of cores.*

– **Batch run** allows to execute projects one after the other. It has been implemented to automate the processing of several projects.

For both modes of operation, the input data have to be read first. If system test has been requested, the data are plotted or saved without header. In the case of data processing, the Interference to Noise Ratio (INR), noise level and eigenvalues are computed. Finally, the output data are computed using the selected method and saved. This operation is summarized in Fig. 6.

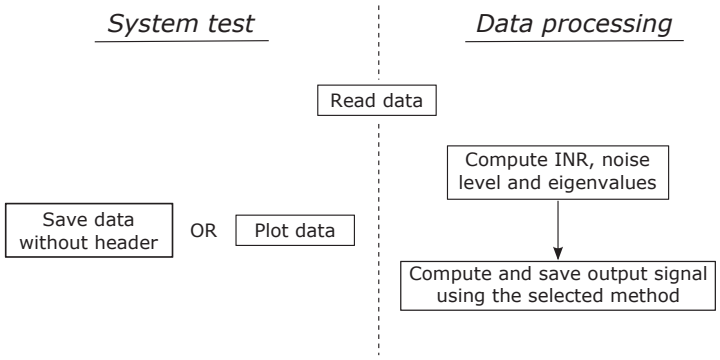

Figure 6: *FiTAT operation. INR stands for Interference to Noise Ratio.*

Before opening the input files to read the data, the filenames are prepared by concatenating the input data generic name and channel names. Then, the data are read and the stripping and/or processing is carried out. Further details regarding the implementation of both modes of operations are provided in Sections 3.2.1 and 3.2.2.

3.2.1 System test

System test allows to either save the acquired data files without header, or to visualize data. In the case where data are saved without header, the data files are opened, read and saved in different files. Because the data files are large, the data have to be read in blocks. Parallel computing has been implemented to increase the efficiency of this operation. If one processor is available, one block of each file is read sequentially until the end of the file. When multiple processors are used, the choice has been made to distribute the files among the processors. Under these assumption, there are two different scenarios: (1) the number of files (no_ch) is a multiple (k) of the number of processors (p), or (2) the number of files cannot be distributed evenly among the available processors. When $no\_ch = k * p$ (scenario 1), k files are attributed to each processor. When $no\_ch = k * p + r$ (scenario 2), k files are attributed to each processor, and the r remaining files are processed afterwards using r processors.

When visualizing the data, the data files are opened, read and plotted. In this case, the data are read and plotted sequentially by blocks of length $Window\_size$. This is a time

consuming operation mostly due to the plotting. In this case, one processor is used because parallel computing is not optimal when plotting.

3.2.2 Data processing

The data processing consists in calculating under constraints a combination of the specified data files. The calculation can be done according to different methods. Details on each method are provided in Section 4.2. The different data files are opened, read and the output is computed by weighting the data of each file. For large data files, the processing is time consuming. As for system test, parallel computing has also been implemented to increase the efficiency of this operation.

When one processor is used, the data files are all processed one block after the other. However when multiple processors are available, the data are distributed over the different processors. Specifically, *no_ch* data files are opened, and one block of data are read by each of the p processors, which means that p consecutive blocks are processed at a time. In this case, the amount of processed data is a multiple of $p * Window\_size$, and the end of the data might not be processed.

3.3 Project data and additional functions

The data of the GUI are saved in a structure type variable named ‘proj_var_struct’. It includes the fields given in Table 1.

Two functions are used to read a project, ‘read_fitat_proj’ and ‘update_gui_field’. ‘read_fitat_proj’ manages the reading of the ‘.prj’ file and stores the extracted values in the ‘proj_var_struct’ variable. ‘update_gui_field’ loads the values of into the GUI. ‘read_fitat_proj’ is in an independant ’.m’ file, and ‘update_gui_field’ is implemented into ’Fitat.m’ (GUI).

Two additional functions are used to save a project, ‘get_proj_var’ and ‘save_fitat_proj’. ‘get_proj_var’ allows to read all the values from the GUI and save them in the variable ‘proj_var_struct’. ‘save_fitat_proj’ write the ‘proj_var_struct’ variable in a ‘.prj’ file (ASCII). ‘get_proj_var’ is implemented into ’Fitat.m’ file (GUI), and ‘save_fitat_proj’ is in a separate ’.m’ file.

The function ‘WriteLogData’ allows to create a log file in the ASCII format to provide the date and time of the analysis, the specified data files, the potential output files, and the different values of the processing parameters (analysis type, window size, etc). It uses the function ‘save_fitat_proj’, and it is implemented in its own ‘.m’ file.

Two others functions are used in FiTAT, ‘report’ and ‘match_subScale’. They are both implemented into ’Fitat.m’ (GUI). ‘report’ allows to print information regarding the data processing into the ‘Acq_proj_print’ component located in the lower part of the GUI. ‘match_-

Table 1: *Fields of the 'proj_var_struct' variable.*

Field name	Data type	Content
dataType	string	current element of 'dataType'
multiProc	integer	value of 'multiProc'
dataFilename	string	'Acq_filename'
chNo_txt	string	'Ch_no'
RefCh	string	current element of 'Ref_ch'
RefElt	integer	current value of 'Ref_ch'
readSize_txt	string	'Window_size' (closest power of 2 is considered)
fileLengthProc	integer	value of 'FileLengthProc'
anaType	string	set to '*syst_test*' or '*data_proc*' depending of the type of requested test
procMeth	string	set to '*pwr_inv*', '*ref_pwr_min*' or '*gen_pwr_min*' depending of the type of requested processing
stripData	integer	value of 'Save_data_ch'
chNo2Save_txt	string	'Ch_no2save'
owr	integer	set to 1 if file overwrite is allowed, set to 0 otherwise

subScale' is called when plotting is requested, and it determines the proper common scale between the different data.

Four functions implemented on their own '.m' file are necessary for FiTAT. 'cut_str' and 'txt2num' are used to isolate values in a string including commas and convert strings to numbers, respectively. Those two functions have been developed at DRDC for other Matlab tools (in particular PAASoM [9]). 'modaldlg' is a GUI called to confirm the closing of the main window and to ask if the user wants to save the data. 'Fitat_closereq' contains the actions to be executed when requesting to close the main window (pushing the close button or clicking on the top right cross button). It is necessary for this function to be on its own '.m' to allow actions when the top right cross button is pushed. The function name has to be specified into 'Fitat.m' (GUI).

4 Structure of the code

When the execute button is pushed in the FiTAT interface, the processing of the data files starts. This section provides details regarding the code which runs depending on the selected mode of operation.

4.1 Description of the most relevant code

The core of the FiTAT tool code is implemented in the function 'Fitat_batch.m'. In this manner, the code is independent from the GUI and it is flexible for batch runs. Only one parameter is required to call this function:

function Fitat_batch(proj_var)

'Proj_var' is a structure variable containing the fields given in Table 1.

At the beginning of the 'Fitat_batch' function, one Matlab environment is set up on each core of the processor unit if multi-processor is used. This is done with the command '*matlabpool open*'. Then, the set of data filenames is prepared and stored in a cell array named 'files'. The output filename is also generated and saved in the variable 'outputFile', and/or the headers are read depending on the selected mode of operation. The sample type is stored in the variable 'SamplType' as a string. Its value is usually 'int16'.

The data file processing starts with the opening of the data files and output file using the command 'fopen'. Then, the data files are read sequentially and by blocks. The data are stored in a matrix called 'read_data'. The columns correspond to the data files. Finally, the 'fileProc' function is called and it returns the processed data 'Vout'. In case multi-processor is used, the output data need to be properly concatenated before being saved. To avoid too many calls of the function reading the data, multiple blocks of data are read at once. In this case, a conversion of the 'read_data' matrix to a cell array is realized and the 'fileProc' function is called for each cell.

At the end of the 'Fitat_batch' function, the data files and output files are closed with the command 'fclose'. If multi-processor was used, the Matlab environments are closed on each processor using the command 'matlabpool close'.

To summarize, the relevant variables are presented in Table 2.

4.2 Implementation of the data processing routines

The data processing of the data itself is done in the function 'fileProc.m'. It takes care of the computation of the processed data according to the selected method. The parameters of

Table 2: *Relevant variables used in 'Fitat_batch.m'.*

Variable name	Data type	Details
files	cell array	data filenames
outputFile	string	processed data filename
SamplType	string	sample data type
read_data	matrix	input data
Vout	vector	processed data

this function are:

function Vout = fileProc(read_data,blockSize,proc_meth,ref_elt)

'read_data' contains the input data matrix, 'blockSize' is the number of samples being processed, 'proc_meth' is the method used to compute the output data, and 'ref_elt' is the reference element.

When real data are processed, the Hilbert function ('hilbert.m', available with the Matlab signal processing toolbox) is called to compute the analytical data. The achieved processing is described below for the different methods. For further information, refer to [10].

4.2.1 Power inversion

The power inversion method allows to steer nulls in the direction of interferences. This method is useful when the desired signals are weak and no *a priori* knowledge of their direction of arrival is available [11]. The formula used to compute the weights of each sensor is:

$$W_{opt} = (I + k\,\phi)^{-1} \cdot T$$

where I is the identity matrix, k is the loop gain, ϕ is the correlation matrix of the incident signals and $T = [1\ 0\ \ldots\ 0]^T$. The length of vector T is equal to the number of channels, and the index equal to 1 corresponds to the reference channel.

To compute the correlation matrix of the incident signals, the interference to noise ratio (INR) and noise level should be extracted from the samples. This is achieved by computing the eigenvalues of the data matrix. The maximum eigenvalue corresponds to the SNR and the minimum one is the noise level. This is implemented using the following Matlab code:

$$\begin{gathered} covmat = transpose(IQdata) * IQdata / length(IQdata(:,1)); \\ [V,D] = eig(covmat); \\ diag\_D\_data = diag(D); \\ noise\_level = min(diag\_D\_data); \\ INR = max(diag\_D\_data) / noise\_level; \end{gathered}$$

IQdata is a matrix of size *Window_size* by *no_ch*. [V,D] = eig(A) produces matrices of eigenvalues (D) and eigenvectors (V) of matrix A, and X = diag(u) puts the elements of vector u on the main diagonal. Then the correlation matrix *RI* and the weights *Wopt* are calculated as follows:

$$\begin{gathered} RI = Vs * Vs' * INR/size(IQdata,1); \\ T = zeros(1,size(IQdata,2))'; \\ T(ref\_elt) = 1; \\ k = 1e9; \\ Wopt = inv(eye(size(IQdata,2)) + k * RI) * T; \end{gathered}$$

Vs corresponds to the eigenvector of the maximum eigenvalue, and ref_elt is the reference element. Y = eye(n) returns the n-by-n identity matrix.

4.2.2 Auxiliary element power miniaturization

This method is also referred to as the Minimum Variance Distortionless Response (MVDR) beamformer and was first derived by Capon [12]. It is sometimes referred to as the Capon beamformer. The MVDR method relies on the power of the desired signals and interferences, and maximizes the output Signal to Noise Ratio. The knowledge of the desired signal is required in this method. The formula used to compute the weights of each sensor is:

$$W_{opt} = \phi^{-1} \cdot U_s$$

where ϕ is the correlation matrix of the incident signals and U_s is the steering vector for the direction of the desired signals. A way to implement this method with real data is to consider the data from one channel, called the reference channel, and the other channel are the auxiliary channel. The power minimization is achieved by a beam formed using the auxiliary channels and subtracted from the reference channel. The weights used to form the beams on the auxiliary channels are then determined by minimizing the mean square value of the difference between the reference channel and the auxiliary channels. To do this, the Matlab built-in function 'lscov' is used:

$$Wopt = -lscov(AuxData, RefData)$$

4.2.3 Generalized power minimization

This method is also referred to as the Minimum Power Distortionless Response (MPDR). It is similar to the MVDR method, except that a different estimate is used to compute the weights:

$$W_{opt} = \frac{{U_s}^H \cdot \phi^{-1}}{{U_s}^H \cdot \phi^{-1} \cdot U_s}$$

where ϕ is the correlation matrix of the incident signals, U_s is the steering vector for the direction of the desired signals, and ^{H} refers to the conjugate transform. Instead of minimizing the mean square value of the difference between the reference antenna output and the output of the auxiliary beam as in the case of the MVDR method, a QR decomposition of the *IQdata* matrix is carried out. The QR decomposition consists in a decomposition of the *IQmatrix* into a product of an orthogonal matrix Q and an upper triangular matrix R. From there, the upper triangular matrix R is used to calculate the weights. No reference channel is required in this method. The corresponding Matlab code is:

$$
\begin{gathered}
c = ones(size(IQdata, 2), 1); \\
[Q, R] = qr(IQdata, 0); \\
g = (R')\_c; \\
Wopt = (R\_g); \\
Wopt = Wopt/(c' * Wopt);
\end{gathered}
$$

The size of the *IQdata* matrix is the number of samples included in a block of data *Window_size* by the number of data files *no_ch*, and $qr(IQdata, 0)$ actually produces the economy-size decomposition. In this case, the first *no_ch* samples of each data file are used for the processing.

5 FiTAT user guide

When opening the FiTAT GUI (or clearing it), the channel name, reference channel, window size and processing length fields are filled with default values, which are read from an ASCII file called ‘DefaultSetup.txt’ (Fig. 7). The user can modify the default values by editing that file.

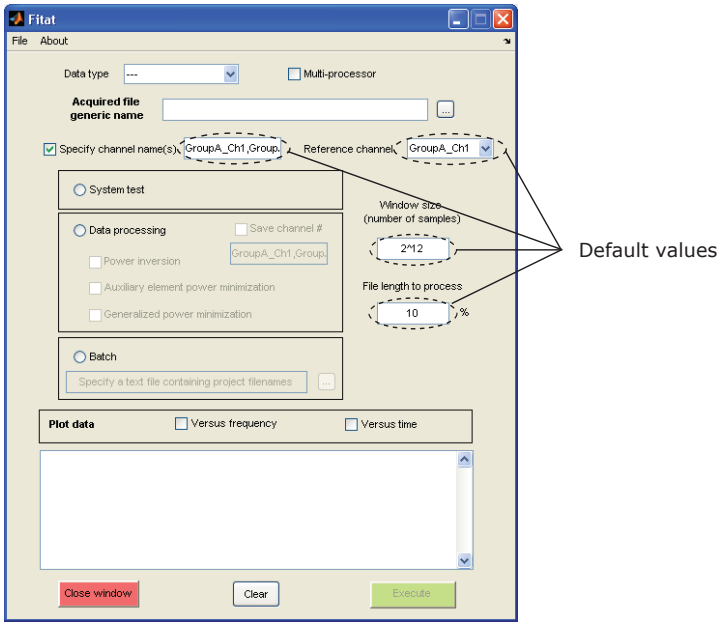

Figure 7: *FiTAT GUI default values.*

The first step to set up a project in FiTAT is to **define the data type**. As shown in Fig. 8, four options are available:

– Gage;
– DTA;
– Real raw data;
– IQ raw data.

Gage and DTA data types should be selected for acquired data, and real and IQ raw data should be chosen for previously stripped or processed data.

Next, the **data generic filename** has to be defined. To do so, the user can use the browse button beside the data generic filename field, and pick one of the files to be processed. After that, the channel name in the data filename has to be replaced with ‘XX’. For instance, if the picked filename is ‘IFcapture_GroupA_Ch1.dat’, ‘GroupA_Ch1’ (part of the filename that varies for a set of data filenames) should be replaced with ‘XX’ (Fig. 9). It is also possible to type (or paste) the filename directly into the GUI. The **channel names** have then to be

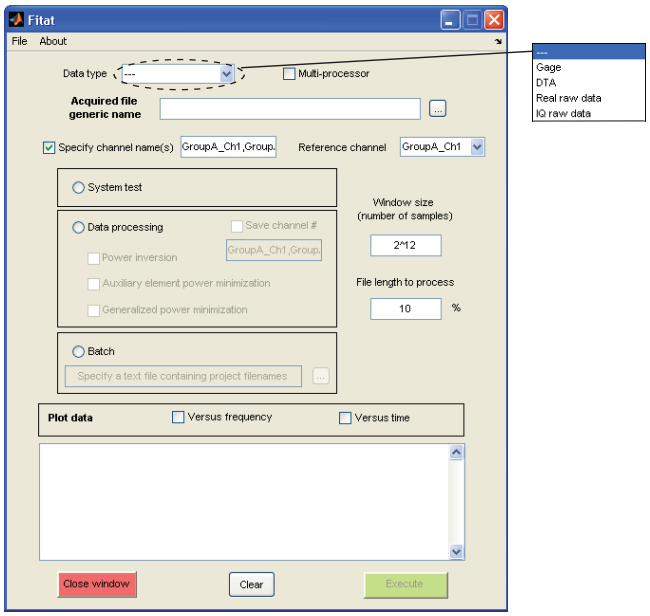

Figure 8: *Data type options.*

specified in the 'Specify channel name(s)' field. A comma is used as separator between channels. The entered channels appear in the **reference channel** field. The user can select the desired reference channel that would be considered for data processing considering the power inversion and auxiliary element power minimization methods.

In addition, the window size and processing length should be specified. The window size corresponds to the number of samples being used to process the data (size of data block). The more rapidly the environment changes, the smaller the value should be. The processing length is the ratio of data included in the data file(s) that will be processed. The user should enter '100' to process the entire file.

Finally, the desired type of processing has to be selected. The three available options are detailed hereafter.

5.1 System test

The system test option can be used to either strip acquired data with the Gage or DTA systems, or visualizing data (acquired and/or previously processed). When dealing with large acquired data files, it is preferable to strip the data first before plotting them (especially with the DTA system). The processing length is very useful in this case to strip and/or plot the data partially.

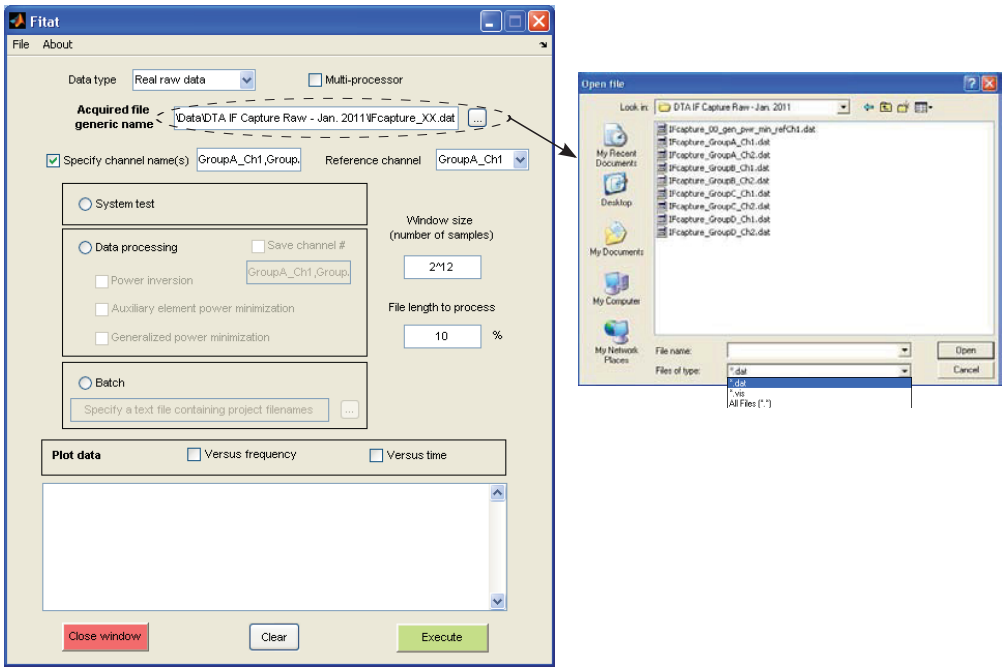

Figure 9: *Definition of the data file generic filename.*

5.2 Data processing

The data processing option allows to process acquired or raw data using several methods:
– power inversion;
– auxiliary element power minimization;
– generalized power minimization.

For details regarding the methods and their implementation, refer to Section 4.2. The auxiliary element and generalized power minimization methods use the specified reference channel. Optionally, stripped acquired data can be saved by checking ’Save channel #’ and specifying the channel names in the frame beside the data processing radiobutton.

5.3 Batch run

When processing very large data file, the computation can be time-consuming. In this case, it can be advantageous to use batch runs. To set up a batch run, the following procedure should be executed:
– Save a project file (.prj) using the ‘File’ menu, and edit it (ascii format) to create other projects (Fig. 10). Comments can be added in the project file using the ‘%’ character in the beginning of each comment line (as in Matlab files);
– Specify the project filenames in a text file, as shown in Fig. 11;

– Check the batch radiobutton in FiTAT, and browse the text file containing the project filenames.

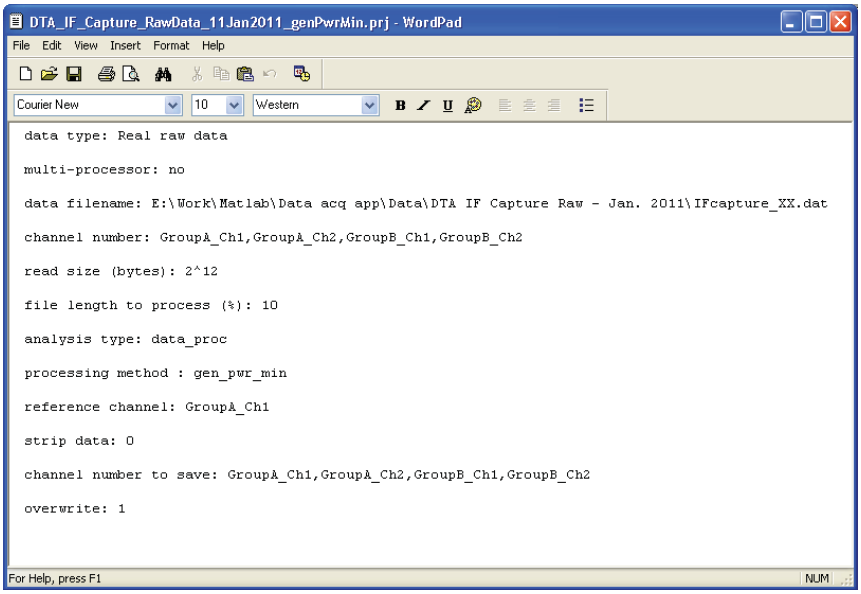

Figure 10: *Project file content.*

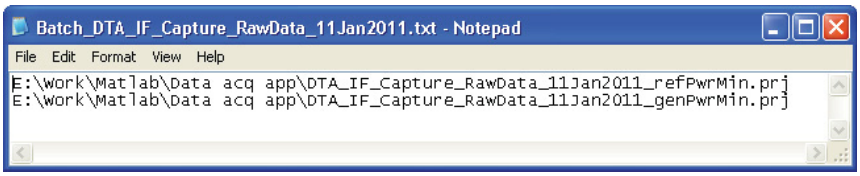

Figure 11: *Batch file content.*

5.4 Result files

The result files are saved in the same directory as the data which corresponds to the path-name of the data generic filename. As a result, it is a good practice to organize the acquired data into folders.

For each simulation, a '.log' file is generated. It includes:
– the filename of the project;
– the date and time at which the computation was performed;
– the values of the variables of the project;
– the filename of the processed data.

In addition to the '.log' file, different files, identified by an extension to the generic name, are saved depending on the selected operation:

– raw data (header removed) - '_raw';
– channel combination - '_combine_operationNumber';
– output data - '_methodName_referenceElement';
– optimal weights - '_Wopt_referenceElement'.

6 Conclusions and perspectives

This report presented the FiTAT tool. The implementation of the GUI and the data handling were described first. In addition, details regarding how data are converted to matrices and the structure of the code were provided. In particular, several nulling algorithms have been implemented. Finally, the instructions to use the FiTAT tool were given. This document is intended to be used as a reference to further develop the FiTAT tool as well as to use this tool.

As for perspectives, FiTAT could be associated to the PAASoM tool [9] for the purpose of studying the radiation characteristics related to the input data. In particular, PAASoM could be fed by steering vectors extracted in FiTAT to compute the array factor and evaluate the radiation characteristics of adaptive antenna arrays. In addition, new algorithms could be implemented, specifically adaptive algorithms and direction finding methods.

References

[1] Gage Octopus 14-bit digitizer, Gage Applied Technologies, Copyright 2009.

[2] D-TA record/playback system (2007), D-TA Systems Inc.

[3] S. McLelland, M. Cowell, M. Clénet, M. Caillet and A. Yasotharan (2010), GPS Anti-Jam Evaluation Testbed (GAJET) Development at Trial Petawawa 09-2, *Defence R&D Canada - Ottawa (DRDC Ottawa TR 2010-118)*.

[4] S. McLelland, M. Cowell and M. Clénet (2010), GAJET-P(Y) at Trial Petawawa 10-1, *Defence R&D Canada - Ottawa (DRDC Ottawa TN 2010-162), internal publication*.

[5] S. McLelland (2010), GPS Anti-Jam Evaluation Testbed (GAJET) Experiments at Trial Fortune Fox, *Defence R&D Canada - Ottawa (DRDC Ottawa TM 2010-192)*.

[6] C. O'Driscoll, D. Borio, M. Petovello, T. Williams and G. Lachapelle (2009), The Soft Approach - A Recipe for a Multi-System, Multi-Frequency GNSS Receiver, *Inside GNSS*, 4(5), 46–51.

[7] Matlab v. R2010b, The MathWorks Inc., Copyright 1984-2010.

[8] S. McLelland, Private communication.

[9] Rogojan, S. and Clénet, M. (2006), Paasom user guide, release 1, *Defence R&D Canada - Ottawa (DRDC Ottawa SL 2006-213)*.

[10] Caillet, M. (2007), Narrowband adaptive antennas - Basic concepts, *Defence R&D Canada - Ottawa (DRDC Ottawa CR 2007-165)*.

[11] Zahm, C. L. (1973), Application of Adaptive Arrays to Supress Strong Jammers in the Presence of Weak Signals, *IEEE Trans. Aerosp. Electron. Syst.*, 9(2), 260–271.

[12] Capon, J. (1969), High-resolution frequency-wavenumber spectrum analysis, *Proc. IEEE*, 57, 1408–1418.

This page intentionally left blank.

DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when document is classified)

1. ORIGINATOR (The name and address of the organization preparing the document. Organizations for whom the document was prepared, e.g. Centre sponsoring a contractor's report, or tasking agency, are entered in section 8.)

Royal Military College of Canada
PO Box 17000, Station Forces
Kingston, Ontario, Canada K7K 7B4

2. SECURITY CLASSIFICATION (Overall security classification of the document including special warning terms if applicable.)

UNCLASSIFIED
(NON-CONTROLLED GOODS)
DMC A
REVIEW: GCEC June 2010

3. TITLE (The complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S, C or U) in parentheses after the title.)

Field Trial data Analysis and Testing (FiTAT) tool

4. AUTHORS (Last name, followed by initials – ranks, titles, etc. not to be used.)

Caillet, M.

5. DATE OF PUBLICATION (Month and year of publication of document.)	6a. NO. OF PAGES (Total containing information. Include Annexes, Appendices, etc.)	6b. NO. OF REFS (Total cited in document.)
October 2011	36	12

7. DESCRIPTIVE NOTES (The category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.)

Contract Report

8. SPONSORING ACTIVITY (The name of the department project office or laboratory sponsoring the research and development – include address.)

Defence R&D Canada - Ottawa
PO Box 17000, Station Forces
Kingston, Ontario, Canada K7K 7B4

9a. PROJECT NO. (The applicable research and development project number under which the document was written. Please specify whether project or grant.)	9b. GRANT OR CONTRACT NO. (If appropriate, the applicable number under which the document was written.)
15en01	A1410FE255

10a. ORIGINATOR'S DOCUMENT NUMBER (The official document number by which the document is identified by the originating activity. This number must be unique to this document.)	10b. OTHER DOCUMENT NO(s). (Any other numbers which may be assigned this document either by the originator or by the sponsor.)
DRDC Ottawa CR 2011-143	

11. DOCUMENT AVAILABILITY (Any limitations on further dissemination of the document, other than those imposed by security classification.)

(X) Unlimited distribution
( ) Defence departments and defence contractors; further distribution only as approved
( ) Defence departments and Canadian defence contractors; further distribution only as approved
( ) Government departments and agencies; further distribution only as approved
( ) Defence departments; further distribution only as approved
( ) Other (please specify):

12. DOCUMENT ANNOUNCEMENT (Any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in (11)) is possible, a wider announcement audience may be selected.)

13. ABSTRACT (A brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual.)

This report presents the Field Trial data Analysis and Testing (FiTAT) tool developed at DRDC Ottawa to analyze and process acquired data files recorded during field trials. Using FiTAT, studies can be conducted to select the appropriate algorithms and hardware meeting cost and performance. The implementation of the FiTAT tool and a user guide are described here to provide useful information to software developers as well as users. In addition, parallel computing is available to reduce the processing time of very large data files.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (Technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus. e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

Adaptive (smart) antennas
Antenna array
Narrow bandwidth

Defence R&D Canada

Canada's leader in Defence
and National Security
Science and Technology

R & D pour la défense Canada

Chef de file au Canada en matière
de science et de technologie pour
la défense et la sécurité nationale

www.drdc-rddc.gc.ca